Automated data-driven tuning of learning-based model predictive control (SelfMPC): A maximum-likelihood approach

Guitao Yang[1], Matteo Scandella[2], Simone Formentin[3], Thomas Parisini[4]

***Abstract*— The practical implementation of Model Predictive Control (MPC) often presents challenges that remain unaddressed in theoretical formulations. Among these challenges, the tuning of the receding horizon cost becomes particularly intricate in the context of data-driven learning-based MPC, where models exhibit partial uncertainty. This paper introduces SelfMPC, a pioneering approach within a Gaussian process learning framework, illustrating that a tracking MPC cost can be formulated as the maximum likelihood estimation of the reference output. This formulation provides automatic cost shaping and effective regularization, eliminating the need for manual tuning efforts. Moreover, the proposed formulation provides a natural way to employ information from empirical experiments into the definition of the MPC optimization problem for unknown systems. Empirical validation against conventional weighting matrix selection methods confirms the effectiveness of the proposed approach.**

I. Introduction

For decades, Model Predictive Control (MPC) [1] has been recognized as a versatile strategy applied across various domains, including industrial processes [2], traffic management [3], [4], biomedical applications [5], and energy management systems [6]. MPC optimizes complex systems in real-time, enhancing performance, safety, and efficiency, but its practical implementation faces significant challenges.

Acquiring an accurate system model for MPC implementation is a fundamental obstacle, with conventional methods sometimes inadequate [7]. Alternative techniques, such as MPC-oriented model calibration [8], [9] or data-driven approaches [10], [11], address this challenge within the realm of Data-driven Predictive Control (DDPC) [12]. Precisely selecting hyperparameters, like weighting matrices and prediction horizons, is also crucial for MPC's effectiveness. Balancing these parameters aligns the control strategy with the given specifications. In scenarios with closed-loop control datasets, inverse optimal control or inverse reinforcement learning [13] can deduce controller preferences. Otherwise, trial-and-error procedures [14] or black-box optimization tools [15] can be used to improve calibration efficiency, but the required closed-loop experiments may not be feasible due to safety or cost concerns.

In this paper, we propose a novel approach for MPC cost function design for reference tracking when the system model is unknown. Leveraging Gaussian process regression [16], [17], we frame cost design as a Maximum Likelihood (ML) Estimation Problem. This approach automatically determines weighting matrices, considers model accuracy, and eliminates the need for additional regularization terms, distinguishing it from other methods like DeePC [18], [19]. We demonstrate the efficacy of this approach, termed SelfMPC, through benchmark simulations.

Organization: In Section II, we formally define the problem and outline the core concept, which is the transformation of a tracking MPC task into a maximum likelihood problem. Section III is devoted to the method designed for the stochastic linear time-invariant setting, while in Section IV, we demonstrate its application within a Bayesian framework. Subsequently, Section V showcases a simulation case study along with a numerical comparison against the prevailing state-of-the-art methods. Some concluding remarks are provided in Section VI.

Notations: We denote by $\mathbb{R}$ and $\mathbb{N}$ the set of real and natural numbers, respectively ($0 \in \mathbb{N}$). Given $n, m \in \mathbb{N}$, $\mathbb{R}^{n\times m}$ is the set of $n \times m$ matrices and $\mathbb{R}^n$ is the set of column vectors of dimension n. Furthermore, $\mathbf{0}_{n\times m}, \mathbf{1}_{n\times m} \in \mathbb{R}^{n\times m}$ are the zero and one matrices, respectively, and $I_n \in \mathbb{R}^{n\times n}$ is the identity matrix. Given $n, i, j \in \mathbb{N}$ and a sequence $x : \mathbb{N} \to \mathbb{R}^n$, $x_i \in \mathbb{R}^n$ is the ith component of x and, if $j \geq i$, $x_{i:j} = (x_h)_{h=i}^{j} \in \mathbb{R}^{n(j-i+1)}$, otherwise $x_{i:j}$ is an empty tuple. With a slight abuse of notation, we use tuples of real numbers and column vectors interchangeably. The matrix $\mathrm{diag}(M_1, M_2, \ldots, M_n)$ is a block diagonal matrix composed of the matrices $M_1, M_2, \ldots, M_n$. Given two matrices A, B, $A \otimes B$ is their Kronecker product. Given a symmetric positive definite matrix $Q \in \mathbb{R}^{n\times n}$ and $x \in \mathbb{R}^n$, $\|x\|_Q^2 = x^\top Qx$

This work has been partially supported by the European Union's Horizon 2020 research and innovation programme under grant agreement no. 739551 (KIOS CoE), by FAIR (Future Artificial Intelligence Research) project, funded by the NextGenerationEU program within the PNRR-PE-AI scheme (M4C2, Investment 1.3, Line on Artificial Intelligence), by the Italian Ministry of Enterprises and Made in Italy in the framework of the project 4DDS (4D Drone Swarms) under grant no. F/310097/01-04/X56, by the PRIN PNRR project P2022NB77E "A data-driven cooperative framework for the management of distributed energy and water resources" (CUP: D53D23016100001), funded by the NextGeneration EU program, and by the Italian Ministry for Research in the framework of the 2017 Program for Research Projects of National Interest (PRIN), Grant no. 2017YKXYXJ.

[1]Guitao Yang is with the Department of Electrical and Electronic Engineering, Imperial College London, London, United Kingdom. He is the corresponding author. Email: `guitao.yang@imperial.ac.uk`

[2]Matteo Scandella was with the Department of Electrical and Electronic Engineering, Imperial College London, London, United Kingdom. He is now with the Department of Management, Information and Production Engineering, University of Bergamo, via Marconi 5, 24044, Dalmine (BG), Italy. Email: `matteo.scandella@unibg.it`

[3]Simone Formentin is with the Department of Electronics, Information and Bioengineering, Politecnico di Milano, via G. Ponzio 34/5, 20133 Milano, Italy. Email: `simone.formentin@polimi.it`

[4]Thomas Parisini is with the Department of Electrical and Electronic Engineering, Imperial College London, London SW7 2AZ, UK, with the Department of Engineering and Architecture, University of Trieste, 34127 Trieste, Italy, and with the KIOS Research and Innovation Center of Excellence, University of Cyprus, CY-1678 Nicosia, Cyprus. Email: `t.parisini@imperial.ac.uk`

and $\|x\|^2 = x^\top x$. For the sake of brevity, we denote with "density" the probability density function. Given a random variable $x \in \mathcal{X}$ with density $\pi : \mathcal{X} \to [0, \infty]$, we write $x \sim \pi(\cdot)$. Given $p \in \mathbb{N}$, $\mu \in \mathbb{R}^p$ and $\Sigma \in \mathbb{R}^{p \times p}$, $\mathcal{N}(\cdot \,|\, \mu, \Sigma)$ denotes the density of the p-dimensional Normal distribution with mean μ and covariance matrix Σ.

II. Problem Statement and Main Idea

In this section, we lay down the main idea behind SelfMPC for the tracking problem of a stochastic dynamical model. In particular, we consider two cases: **(a)** when the stochastic property of the system to control is known for each time instant, and **(b)** when the system is not known and need to be estimated from data. For both cases, we consider a general discrete causal stochastic model with $n_u \in \mathbb{N}$ inputs and $n_y \in \mathbb{N}$ outputs that maps control variable sequences $u : \mathbb{N} \to \mathbb{R}^{n_u}$ to measurements sequences $y : \mathbb{N} \to \mathbb{R}^{n_y}$ and variable to control $z : \mathbb{N} \to \mathbb{R}^{n_z}$ according to

$$\forall t \in \mathbb{N}, \qquad y_t, z_t \big| y_{0:t-1}, u_{0:t}, \theta \sim \psi_t\big(\cdot \,\big|\, y_{0:t-1}, u_{0:t}, \theta\big), \tag{1}$$

where $n_u \in \mathbb{N}$ is the number of inputs, $n_y \in \mathbb{N}$ is the number of measurements, $n_z \in \mathbb{N}$ is the number of variables to control and $\theta \in \Theta$. In particular, for every $t \in \mathbb{N}$, $\psi_t : \mathbb{R}^{n_y} \times \mathbb{R}^{n_z} \times \mathbb{R}^{tn_y} \times \mathbb{R}^{(t+1)n_u} \times \Theta \to [0, \infty]$ is the joint density of the couple (y_t, z_t) conditioned on the past measurements and inputs. Therefore, $\psi_0 : \mathbb{R}^{n_y} \times \mathbb{R}^{n_z} \times \mathbb{R}^{n_u} \times \Theta \to [0, \infty]$ is the density of the initial values of the measurements and control variables conditioned on the initial value of the input. Additionally, θ is a parameter that describes the behavior of the model. The availability of θ is the main difference between Case **(a)** and **(b)**. In particular, we assume that the knowledge of θ is available only for Case **(a)**. Instead, the functions ψ_t are assumed to be known in both cases, for every $t \in \mathbb{N}$.

Given a time horizon $m \in \mathbb{N}$, the aim of the paper is to devise a procedure that, for every $t \in \mathbb{N}$, selects the input $u_{t:t+m} \in \mathbb{R}^{n_u(m+1)}$ in order to shape the control variable trajectory $z_{t+1:t+m} \in \mathbb{R}^{n_z m}$ to be as close as possible to a reference trajectory $r_{t+1:t+m} \in \mathbb{R}^{n_z m}$. In particular, we employ the receding horizon approach to track a certain reference signal. Classical stochastic MPC approaches select the input by minimizing the expected value of a loss function subject to constraints that model the dynamic of the model at each time instant. Instead, we propose to select the input that maximizes the likelihood that the future trajectory of the variable to control $z_{t+1:t+m}$ is equal to the reference trajectory $r_{t+1:t+m}$.

In Case **(a)**, SelfMPC provides a way to automatically select the function to maximize. However, the main advantage of the proposed method is the ability to be easily adapted to the data-driven framework where the model is not known, like in Case **(b)**.

A. The case with a known system

Let us first consider the case where θ is known. Since we aim to maximize the likelihood of the future variable to control $z_{t+1:t+m}$, for every $t \in \mathbb{N}$, we need to define its density conditioned on the known measurements and parameters at time t. In particular, using the model defined in (1), this density is defined as

$$\Psi_{m|t}\big(z_{t+1:t+m} \,\big|\, y_{0:t}, u_{0:t-1}, u_{t:t+m}, \theta\big) := \int_{\mathbb{R}^{mn_y}} \prod_{\tau=t+1}^{t+m} \psi_t\big(y_\tau, z_\tau \,\big|\, y_{0:\tau-1}, u_{0:\tau}, \theta\big)\, dy_{t+1:t+m}, \tag{2}$$

for every $m \in \mathbb{N} \setminus \{0\}$ and $t \in \mathbb{N}$. The function $\Psi_{m|t} : \mathbb{R}^{mn_z} \times \mathbb{R}^{(t+1)n_y} \times \mathbb{R}^{tn_u} \times \mathbb{R}^{(m+1)n_u} \times \Theta \to [0, \infty]$ describes the likelihood of the trajectory $z_{t+1:t+m}$ given

- $y_{0:t} \in \mathbb{R}^{(t+1)n_y}$, the measurements of the outputs available after collecting the measurement at time t,
- $u_{0:t-1} \in \mathbb{R}^{tn_u}$, the control input variables available after collecting the measurement at time t,
- $u_{t:t+m} \in \mathbb{R}^{(m+1)n_u}$, the trajectory of the inputs that is applied between the current time t and the end of the prediction window $t + m$,
- $\theta \in \Theta$, the known parameter that define the model.

Therefore, the SelfMPC tracking problem is solved by selecting the input sequence $u_{t:t+m}$ that maximizes the density defined in (2). More formally, the selected control variable is the solution of

$$\max_{u_{t:t+m}} \; \Psi_{m|t}\big(r_{t+1:t+m} \,\big|\, y_{0:t}, u_{0:t-1}, u_{t:t+m}, \theta\big).$$

From now on, we will refer to this approach as SelfMPC.In Section III, we analyze this method in the case of LTI models with Gaussian noise.

B. The case with an estimated system (learning-based MPC)

In this section, we consider the case where the parameter θ is not known, and the control input variable has to be decided purely from the available data. In this case, we cannot rely on the density $\Psi_{m|t}$ defined in (2) since θ is not known. Instead, we need to compute the density of the variable to control given only the available data. Therefore, for every $m \in \mathbb{N} \setminus \{0\}$ and $t \in \mathbb{N}$, we consider

$$\widehat{\Psi}_{m|t}\big(z_{t+1:t+m} \,\big|\, y_{0:t}, u_{0:t-1}, u_{t:t+m}\big) := \int_{\Theta} \Psi_{m|t}\big(z_{t+1:t+m} \,\big|\, y_{0:t}, u_{0:t-1}, u_{t:t+m}, \theta\big) \cdot \\ \cdot\, \Phi_t\big(\theta \,\big|\, y_{0:t}, u_{0:t-1}\big)\, d\theta,$$

where $\Phi_t : \Theta \times \mathbb{R}^{(m+1)n_y} \times \mathbb{R}^{mn_y} \to \mathbb{R}$ is the density of the parameter θ given the available measurements at time t, and $\widehat{\Psi}_{m|t} : \mathbb{R}^{mn_z} \times \mathbb{R}^{(t+1)n_y} \times \mathbb{R}^{tn_u} \times \mathbb{R}^{(m+1)n_u} \to \mathbb{R}$ is the density of the next m samples of the variable to control given the available measurements at time t. In particular, the density Φ_t is defined by the method used for the identification of the parameter θ given the available data. Note that the parameter θ is conditioned only on the past measurements because we assumed that the system is causal.

Then, the tracking problem with unknown θ is solved by selecting the input sequence as the one maximizing the conditional density of obtaining the desired output trajectory given the available data. More formally, we have

$$\max_{u_{t:t+m}} \quad \widehat{\Psi}_{m|t}(r_{t+1:t+m} \,|\, y_{0:t}, u_{0:t-1}, u_{t:t+m})$$

From now on, we refer to this approach as the data-driven SelfMPC. In Section IV, we illustrate the detailed implementation of SelfMPC in the case of a linear model with unknown parameters and additive Gaussian noise.

III. SelfMPC for Linear Systems

In this section, we implement the proposed SelfMPC method on a finite-dimensional LTI system with Gaussian noise, given by

$$\begin{aligned} y_{t+1} &= Ay_t + Bu_t + Ew_t, && \text{(3a)} \\ z_t &= Cy_t + Du_t + Fw_t, \quad \forall t \in \mathbb{N} && \text{(3b)} \end{aligned}$$

where $A \in \mathbb{R}^{n_y \times n_y}$, $B \in \mathbb{R}^{n_y \times n_u}$, $E \in \mathbb{R}^{n_y \times n_w}$, $C \in \mathbb{R}^{n_z \times n_y}$, $D \in \mathbb{R}^{n_z \times n_u}$, $F \in \mathbb{R}^{n_z \times n_w}$, $w_t \in \mathbb{R}^{n_w}$ is the noise vector that affect the system at time t, and $n_w \in \mathbb{N}$ is the number of noises affecting the system. Then we consider the following stochastic assumption on the stochastic process w.

Assumption 1: The stochastic process w is a white process and, for every $t \in \mathbb{N}$, $w_t \sim \mathcal{N}\big(\cdot \,\big|\, \mathbf{0}_{n_w \times 1}, W\big)$ where $W \in \mathbb{R}^{n_w \times n_w}$ is a positive definite matrix.

From this assumption, and the fact that a linear combination of Normal random variables is a Normal random variable, the measurements y_t and the variables to control z_t are Normal random variables for every $t \in \mathbb{N}$. Furthermore, using the definition in (3), we obtain that the stack of m future variable to control, for all $t \in \mathbb{N}$, is given by

$$z_{t+1:t+m} = P_m y_t + Q_m u_{t:t+m} + R_m w_{t:t+m} \qquad (4)$$

where $P_m \in \mathbb{R}^{mn_z \times n_y}$, $Q_m \in \mathbb{R}^{mn_z \times (m+1)n_u}$ and $R_m \in \mathbb{R}^{mn_z \times (m+1)n_w}$ are block matrices where the i-th block of P_m is CA^i for all $i \in \{1, \ldots, m\}$ $(j = 1)$. Moreover, the (i, j)-th blocks of Q_m and R_m are given by

$$(i,j)\text{-th block of } Q_m \Rightarrow \begin{cases} CA^{i-j}B & i \geq j \\ D & i+1 = j \\ \mathbf{0}_{n_z \times n_u} & i+1 < j \end{cases},$$

$$(i,j)\text{-th block of } R_m \Rightarrow \begin{cases} CA^{i-j}E & i \geq j \\ F & i+1 = j \\ \mathbf{0}_{n_z \times n_w} & i+1 < j \end{cases},$$

for all $i \in \{1, \ldots, m\}$ and $j \in \{1, \ldots, m+1\}$.

A. Prediction of the controlled variable

From (4), we notice that

$$z_{t+1:t+m}\big|y_{0:t}, u_{0:t-1}, u_{t:t+m} = z_{t+1:t+m}\big|y_t, u_{t:t+m}.$$

Additionally, $z_{t+1:t+m}$ is a linear combination of deterministic variables and random variables distributed according to a Normal distribution. Thus, $z_{t+1:t+m}$ is a Normal variable defined as

$$z_{t+1:t+m}\big|y_t, u_{t:t+m} \sim \mathcal{N}\big(\cdot \,\big|\, \mu_m(y_t, u_{t:t+m}), \Sigma_m(y_t, u_{t:t+m})\big) \qquad (5)$$

where the mean $\mu_m : \mathbb{R}^{n_y} \times \mathbb{R}^{(m+1)n_u} \to \mathbb{R}^{mn_z \times 1}$ and variance $\Sigma_m : \mathbb{R}^{n_y} \times \mathbb{R}^{(m+1)n_u} \to \mathbb{R}^{mn_z \times n_z}$ are determined by the current measurement and the future input. Then, from (4) and Assumptions 1, we obtain

$$\mu_m(y_t, u_{t:t+m}) = P_m y_t + Q_m u_{t:t+m}, \quad \Sigma_m = R_m \overline{W}_m R_m^\top,$$

where $\overline{W}_m := I_{m+1} \otimes W \in \mathbb{R}^{(m+1)n_w \times (m+1)n_w}$. Note that the variance matrix Σ_m is invariant with respect to the current measurement y_t or the future control input $u_{t:t+m}$. For this reason, and for simplicity, we omit its argument.

B. Maximum Likelihood Algorithm

In this part, we introduce the detailed algorithm by deploying the SelfMPC approach in this case. Referring to the general case analyzed in Section II-A and using (5), we obtain that

$$\begin{aligned} &\Psi^{m|t}\big(r_{t+1:t+m} \,\big|\, y_{0:t}, u_{0:t-1}, u_{t:t+m}, \theta\big) \\ &\qquad = \mathcal{N}\big(r_{t+1:t+m} \,\big|\, \mu_m(y_t, u_{t:t+m}), \Sigma_m\big) \end{aligned}$$

where $r_{t+1:t+m}$ represents the desired trajectory for the next m time steps. Therefore, the optimization variable is selected by solving the optimization problem

$$\max_{u_{t:t+m}} \quad \mathcal{N}\big(r_{t+1:t+m} \,\big|\, \mu_m(y_t, u_{t:t+m}), \Sigma_m\big).$$

However, for computational reason, it is convenient to equivalently minimize its logarithm. Then, using (5), we obtain

$$\min_{u_{t:t+m}} \quad \delta_1(u_{t:t+m})^\top \, \Sigma_m^{-1} \, \delta_1(u_{t:t+m})$$

with $\delta_1(u_{t:t+m}) := r_{t+1:t+m} - \mu_m(y_t, u_{t:t+m})$.

Please observe that the cost function exhibits a structure akin to conventional MPC. It consists of a reference matching cost, along with a component that factors in the control input's aggressiveness. However, the crucial distinction lies in the automatic selection of optimal weighting factors, derived from the maximum likelihood approach, obviating the need for arduous trial-and-error procedures. It is worth noting that in real-world applications, these weight matrices are typically constrained to be diagonal to minimize the tuning effort, albeit at the cost of potentially suboptimal results. In this case, such a constraint is not necessary.

IV. SelfMPC for data-driven FIR models

In this section, we illustrate the implementation of SelfMPC under a data-driven scenario. More specifically, we consider the following finite impulse response (FIR) linear system

$$\begin{aligned} y_{t+1} &= f_L\big(u_{t-n_s+1:t}\big) + Ew_t, \\ z_t &= Gy_t + Hu_t + Kw_t, \quad \forall t \in \mathbb{N} \end{aligned}$$

where $n_s \in \mathbb{N}$, $n_w \in \mathbb{N}$, $f_L : \mathbb{R}^{n_u n_s} \to \mathbb{R}^{n-y}$ is a linear mapping, $E \in \mathbb{R}^{n_y \times n_w}$, $G \in \mathbb{R}^{n_z \times n_y}$, $H \in \mathbb{R}^{n_z \times n_u}$ and $K \in \mathbb{R}^{n_z \times n_w}$. In this scenario, the Assumption 1 is preserved. Here, we assume that the function f_L is unknown. Instead, n_s and the matrices G, H, K, E and W are considered known. Furthermore, for every $t \in \mathbb{N}$, we assume that there are $T \in \mathbb{N}$ measurements available with $T \geq n_s$. In particular, we suppose to know the vectors $y_{t-T+1:t}$ and $u_{t-T-n_s+1:t}$ for every $t \in \mathbb{N}$.

A. Prediction of the controlled variable

We adopt the finite impulse response description of the system, which gives

$$y_t^{i_y} = \sum_{k=1}^{n_s} \sum_{i_u=1}^{n_u} g_{i_y,i_u,k}\, u_{t-k}^{i_u} + e_{i_y} w_t$$

where, for all $t \in \mathbb{N}$, $i_u \in \{1, \dots, n_u\}$ and $i_y \in \{1, \dots, n_u\}$, $e_{i_y} \in \mathbb{R}^{1 \times n_w}$ is the i_y-th row of matrix E, $y_t^{i_y} \in \mathbb{R}$ is the i_y-th element of y_t, $u_{t-k}^{i_u}$ is the i_u-th element of u_t and $(g_{i_y,i_u,k})_{k=1}^{n_s}$ are unknown coefficients that defines the linear function f_L. Therefore, the relation between the last T available data of the i_y-th measurement $Y_t^{i_y} := y_{(t-T+1):t}^{i_y} \in \mathbb{R}^T$, and the past inputs $u_{(t-T-n_s+1):(t-1)}$ is given by

$$Y_t^{i_y} = \phi_t \theta^{i_y} + E_T^{i_y} w_{t-T:t-1},$$

where $E_T^{i_y} = I_T \otimes e_{i_y} \in \mathbb{R}^{T \times T n_w}$,

$$\phi_t := \begin{bmatrix} u_{t-T}^\top & u_{t-T-1}^\top & \cdots & u_{t-T-n_s+1}^\top \\ \vdots & \vdots & \ddots & \vdots \\ u_{t-1}^\top & u_{t-2}^\top & \cdots & u_{t-n_s}^\top \end{bmatrix} \in \mathbb{R}^{T \times n_u n_s},$$

$$\theta^{i_y} := \left[g_{i_y,1,1}, \dots, g_{i_y,1,n_u}, g_{i_y,2,1}, \dots, g_{i_y,s n_u}\right]^\top \in \mathbb{R}^{n_u n_s}.$$

Then, the relation between all the past T measurements $y_{t-T+1:t}^{1:n_y}$ with the past inputs $u_{(t-T-n_s+1):(t-1)}$ is given by $Y_t = \Phi_t \theta + E_T \mathbf{w}_t$ where

$$\begin{aligned}
Y_t &:= \left[\left(Y_t^1\right)^\top, \left(Y_t^2\right)^\top, \dots, \left(Y_t^{n_y}\right)^\top\right]^\top \in \mathbb{R}^{T n_y}, \\
\Phi_t &:= I_{n_y} \otimes \phi_t \in \mathbb{R}^{T n_y \times n_s n_u n_y}, \\
\theta &:= \left[\left(\theta^1\right)^\top, \left(\theta^2\right)^\top, \dots, \left(\theta_y^n\right)^\top\right]^\top \in \mathbb{R}^{n_s n_u n_y \times 1}, \\
E_T &:= \operatorname{diag}\left(E_T^1, E_T^2, \dots, E_T^{n_y}\right) \in \mathbb{R}^{T n_y \times T n_w n_y}, \\
\mathbf{w}_t &:= \mathbf{1}_{1 \times n_y} \otimes w_{t-T:t-1} \in \mathbb{R}^{T n_w n_y \times n_y}.
\end{aligned}$$

Similarly, the relation between the future measurements in the time window considered by the controller and the input variable is given by

$$\hat{Y}_t = \hat{\Phi}_t(u_{t:t+m-1})\theta + E_m \hat{\mathbf{w}}_t \tag{6}$$

where

$$\begin{aligned}
\hat{Y}_t &:= \left[\left(\hat{Y}_t^1\right)^\top, \left(\hat{Y}_t^2\right)^\top, \dots, \left(\hat{Y}_t^{n_y}\right)^\top\right]^\top \in \mathbb{R}^{m n_y}, \\
\hat{\Phi}_t &:= I_{n_y} \otimes \hat{\phi}_t \in \mathbb{R}^{m n_y \times n_s n_u n_y}, \\
E_m &:= \operatorname{diag}\left(I_m \otimes e_1, \dots, I_m \otimes e_{n_y}\right) \in \mathbb{R}^{m n_y \times m n_w n_y}, \\
\hat{\mathbf{w}}_t &:= \mathbf{1}_{1 \times n_y} \otimes w_{t:t+m-1} \in \mathbb{R}^{m n_w n_y \times n_y},
\end{aligned}$$

in which, for every $i_y \in \{1, \dots, n_y\}$, $\hat{Y}_t^{i_y} := y_{t+1:t+m}^{i_y} \in \mathbb{R}^T$ and $\hat{\phi}_t \in \mathbb{R}^{m \times n_u n_s}$ is the matrix whose (i,j)th element is u_{t+j-i}. Note that $\hat{\Phi}_t$ is a function of $u_{t:t+m-1}$. By defining $\hat{Z}_t := z_{t+1:t+m}$, from (6), we write

$$\begin{aligned}
\hat{Z}_t &= G_m \Gamma \hat{Y}_t + H_m u_{t+1:t+m} + K_m \hat{\mathbf{w}}_t \\
&= G_m \Gamma \hat{\Phi}_t \theta + G_m \Gamma E_m \hat{\mathbf{w}}_t + H_m u_{t+1:t+m} + K_m \hat{\mathbf{w}}_t
\end{aligned}$$

where $G_m := I_m \otimes G$, $H_m := I_m \otimes H$, $K_m := I_m \otimes K$, and Γ denotes the permutation matrix, such that $\Gamma \hat{Y}_t = y_{t+1:t+m}$.

We employ a Bayesian approach to derive the cost function to maximize. Therefore, we consider a prior on the unknown vector θ. In particular, we assume $\theta \sim \mathcal{N}\left(\cdot \,\middle|\, \mathbf{0}_{n_s n_u n_y \times 1}, \Sigma_\theta\right)$ where $\Sigma_\theta \in \mathbb{R}^{n_s n_u n_y \times n_s n_u n_y}$ is a valid covariance matrix. The selection of Σ_θ is extensively studied in the literature [16], and multiple interesting choices are available, i.a. stable splines [20], [21] or DC kernel [22].

To formulate the density of the parameters given the data available, we first note that, according to Assumption 1, we have $\mathbf{w}_t \sim \mathcal{N}\left(\cdot \,\middle|\, \mathbf{0}_{T n_y n_w \times 1}, W_T\right)$ and $\hat{\mathbf{w}}_t \sim \mathcal{N}\left(\cdot \,\middle|\, \mathbf{0}_{m n_y n_w \times 1}, W_m\right)$ where $W_T := I_{T n_y} \otimes W$ and $W_m := I_{m n_y} \otimes W$. Now, we can write the joint density between the available measurements Y_t and the variable to control $\hat{Z}_t$ marginalizing θ. Using the properties of the Normal distribution, we obtain

$$\begin{bmatrix} Y_t \\ \hat{Z}_t \end{bmatrix} \sim \mathcal{N}\left(\cdot \,\middle|\, \begin{bmatrix} \mu_{Y_t} \\ \mu_{\hat{Z}_t} \end{bmatrix}, \begin{bmatrix} \Sigma_{Y_t} & \Sigma_{Y_t \hat{Z}_t} \\ \Sigma_{Y_t \hat{Z}_t}^\top & \Sigma_{\hat{Z}_t} \end{bmatrix}\right)$$

where

$$\begin{aligned}
&\mu_{Y_t} := \mathbf{0}_{T n_y \times 1}, \qquad \mu_{\hat{Z}_t} := H_m u_{t+1:t+m}, \\
&\Sigma_{Y_t} := \Phi_t \Sigma_\theta \Phi_t^\top + E_T W_T E_T^\top, \quad \Sigma_{Y_t \hat{Z}_t} := \Phi_t \Sigma_\theta \hat{\Phi}_t^\top \Gamma^\top G_m^\top, \\
&\Sigma_{\hat{Z}_t} := G_m \Gamma \hat{\Phi}_t \Sigma_\theta \hat{\Phi}_t^\top \Gamma^\top G_m^\top + K_m W_m K_m^\top \\
&\qquad\qquad + G_m \Gamma E_m W_m E_m^\top \Gamma^\top G_m^\top.
\end{aligned}$$

Hence, from the properties of the Normal distribution, we can derive the density of the future variable to control $\hat{Z}_t$ conditioned on the collected past measurements Y_t. In particular, we have $\hat{Z}_t | Y_t \sim \mathcal{N}\left(\cdot \,\middle|\, \mu_{\hat{Z}_t|Y_t}, \Sigma_{\hat{Z}_t|Y_t}\right)$ where

$$\begin{aligned}
\mu_{\hat{Z}_t|Y_t} &:= \mu_{\hat{Z}_t} + \Sigma_{Y_t \hat{Z}_t}^\top \Sigma_{Y_t}^{-1} Y_t, \\
\Sigma_{\hat{Z}_t|Y_t} &:= \Sigma_{\hat{Z}_t} - \Sigma_{Y_t \hat{Z}_t}^\top \Sigma_{Y_t}^\top \Sigma_{Y_t \hat{Z}_t}.
\end{aligned}$$

Both $\mu_{\hat{Z}_t|Y_t}$ and $\Sigma_{\hat{Z}_t|Y_t}$ depend on $\hat{\Phi}_t$ that is a function of $u_{t:t+m}$. Therefore, the control variable has the effect to modify both the expected value and the variance of the density of the prediction of the future variables to control.

B. Maximum Likelihood Algorithm

Similar to the formulation in Section III-B, we solve the following optimization problem

$$\begin{aligned}
\min_{u_{t:t+m}} \; & \delta_2(u_{t:t+m})^\top \left(\Sigma_{\hat{Z}_t|Y_t}(u_{t:t+m-1})\right)^{-1} \delta_2(u_{t:t+m}) \\
& + \log\det\left(\Sigma_{\hat{Z}_t|Y_t}(u_{t:t+m-1})\right)
\end{aligned}$$

where $\delta_2(u_{t:t+m}) := r_{t+1:t+m} - \mu_{\hat{Z}_t|Y_t}(u_{t:t+m})$.

The data-driven SelfMPC cost function comprises two terms. The first minimizes the expected prediction to match the desired trajectory, weighted by the inverse prediction variance. Instead, The second acts as a regularizer, penalizing high-variance input sequences to avoid uncertain model behavior. Importantly, this cost function, like the one discussed in Section III-B, requires no additional tuning.

V. Numerical example

In this section, we conduct a comprehensive performance analysis of the proposed SelfMPC method on a benchmark numerical example originally introduced in [23, Sec. 7.1]. In addition, we consider the controlled variable z_t to be a linear transformation of the measurements, resulting in the following dynamic system model:

$$y_{t+1} = \underbrace{\begin{bmatrix} 0.7326 & -0.0861 \\ 0.1722 & 0.9909 \end{bmatrix}}_{A} y_k + \underbrace{\begin{bmatrix} 0.0609 \\ 0.0064 \end{bmatrix}}_{B} u_k + w_t \quad (7a)$$

$$z_t = \underbrace{\begin{bmatrix} 1 & 1 \end{bmatrix}}_{C} y_t + \underbrace{\begin{bmatrix} 0.5 & 0.5 \end{bmatrix}}_{F} w_t \quad (7b)$$

where the stochastic influence w_t is represented by a Gaussian noise vector, with each of its elements having a standard deviation of 5×10^{-2}.

We proceed by evaluating the performance of the SelfMPC strategy under two distinct scenarios. First, we assume a perfect knowledge of the system model and compare its performance against conventional MPC using manually tuned control gains. This comparison serves to highlight the efficacy of our proposed approach at automatically selecting the cost function. Subsequently, we implement the SelfMPC within a data-driven framework and compare its control performance with that of the regularized DeePC (r-DeePC) method [24]. This comparison is instrumental in demonstrating the advantages inherent in our designed data-driven control strategy.

Scenario 1 (Known model): In this scenario, we assume the designer possesses full knowledge of the model parameters. To compare our proposed SelfMPC with conventional Model Predictive Control (MPC), we seek the solution to the following optimization problem at each time step t:

$$\begin{aligned} \min_{u_{0:m-1}} \quad & \|\bar{z}_m - r_{k+m}\|^2 + \sum_{k=1}^{m-1} \|\bar{z}_k - r_{k+t}\|_Q^2 + \|u_k\|_R^2 \\ \text{s.t.} \quad & \bar{z}_k = C\bar{y}_k \qquad\qquad\quad k \in \{1, 2, \dots, m\} \\ & \bar{y}_{k+1} = A\bar{y}_k + Bu_k \quad k \in \{0, 2, \dots, m-1\} \\ & -5 \le u_k \le 5 \qquad\quad\; k \in \{0, 1, \dots, m-1\} \\ & \bar{y}_0 = y_t \end{aligned}$$

Here, we set the prediction horizon $m = 10$, and the gains are manually tuned as $Q = P = 1$ and $R = 0.01$. We consider a constant reference of $r_t = 1$, for every $t \in \mathbb{N}$. Subsequently, we implement the SelfMPC as described in Section III, which autonomously generates the control gains. With an identical prediction horizon, m, and input variable bounds we compare the performance of these two control strategies using the performance index $T^{-1}\|z_t - r_t\|^2$. Through the examination of 100 different realizations, the performance

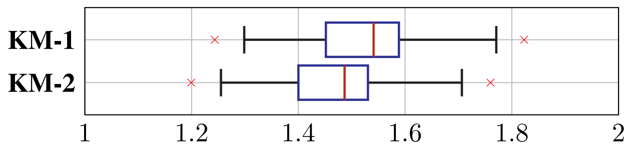

Fig. 1. Box plots of the control performance index on the controlled variable z_t over 100 Monte Carlo simulations in Scenario 1.

of both methods is illustrated in Fig. 1 where the examined methods are listed as

KM-1. Conventional MPC with tuned gains.
KM-2. SelfMPC with automatically generated gains.

The analysis presented in Fig. 1 reveals that SelfMPC exhibits a marginally superior control performance in comparison to conventional MPC with manually tuned gains. This advantage arises from the automatic selection of gains by the SelfMPC approach, thereby eliminating the requirements for extensive experimentation on the system.

Scenario 2 (Data-driven): In this scenario, we address the case where the designer lacks prior knowledge of the model parameters. Here, we compare the proposed approach with the regularized DeePC (r-DeePC) method [25] that was recently developed to handle similar problems. For the sake of comparison, we implement r-DeePC with both a set of randomly selected gains and a set of manually tuned gains.

For the data-driven SelfMPC method, we follow the algorithm as described in Section IV-B. The matrix Σ_θ is the symmetric semi-positive definite matrix whose (i, j)-th element is $\exp(-\beta \max(i, j))$ where $\beta \in [0, \infty)$ is a parameter to tune. Since both methods rely on a previously collected set of data, we collect $N = 80$ measurements from the plant fed with a white noise uniformly distributed between -5 and 5. The collected dataset is influenced by additive noise, as stipulated in (7). Regarding the proposed method, the noise variance $W \in [0, \infty)$ and $\beta \in [0, \infty)$ are tuned using the empirical Bayes approach [26, Sec. 5.4.1] using the T data available at time 0.

Furthermore, we investigate the impact of different prediction horizons, considering both a shorter horizon ($m = 10$) and a longer horizon ($m = 20$). As a reference point, we include the performance of the Oracle MPC, as defined in Scenario 1 with manually tuned gains, under the assumption of perfect knowledge of the system model. The comprehensive evaluation of the controlled variable performance, measured using the index used in Scenario 1, is visualized in Fig. 2 where the examined methods with different settings are listed as

DD-1. Oracle MPC given model with the tuned gains.
DD-2. r-DeePC with random gains and $m = 10$.
DD-3. r-DeePC with tuned gains and $m = 20$.
DD-4. r-DeePC with tuned gains and $m = 10$.
DD-5. Data-driven SelfMPC with $m = 10$.
DD-6. Data-driven SelfMPC with $m = 20$.

From the results presented in Fig. 2, we observe that the performance of r-DeePC is significantly influenced by the choice of gain values. The distinction between properly

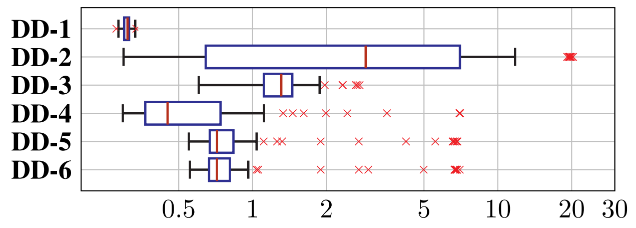

Fig. 2. Box plots (logarithm x-axis) of the control performance index on the controlled variable z_t over 100 Monte Carlo simulations in Scenario 2.

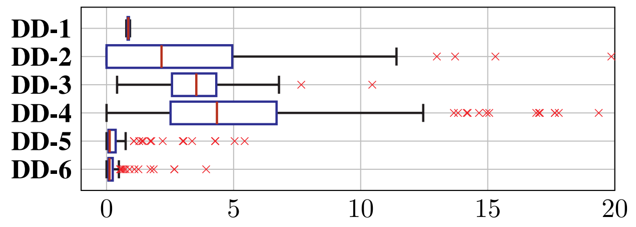

Fig. 3. Box plots of the variance of the input variable u_t over 100 Monte Carlo simulations in Scenario 2.

tuned gains and randomly selected gains is quite evident. In contrast, the data-driven SelfMPC method demonstrates a performance level comparable to r-DeePC with manually tuned gains, while avoiding the need for experimentation on the system. Notably, the performance of r-DeePC deteriorates as the prediction horizon increases, whereas the data-driven SelfMPC remains resilient to this parameter change. Another interesting property of the proposed SelfMPC is the behavior of the selected control variable. In Fig. 3, we plot the variance of the control variables in the considered cases. Here, we can note that the input sequence generated by r-DeePC exhibits higher variance, indicating significant input oscillations. Conversely, the input sequence produced by the data-driven SelfMPC is notably smoother, a characteristic particularly advantageous for the actuator usage in real-world applications.

VI. Conclusions

MPC, a powerful control method, spans diverse fields but faces challenges in precise system modeling, hyperparameter selection, and complex calibration. To address the above issues, we introduce SelfMPC, which selects control variables by maximizing the likelihood of obtaining the desired reference. This approach automatically tunes the cost function and enables data-driven control via Gaussian process regression for model learning, eliminating additional regularization tuning and the need of closed-loop experiments.

Future work involves extending SelfMPC to nonlinear plant control and exploring real-world applications.

References

[1] F. Borrelli, A. Bemporad, and M. Morari, *Predictive control for linear and hybrid systems*. Camb. Univ. Press, 2017.

[2] M. L. Darby and M. Nikolaou, "MPC: Current practice and challenges," *Control Eng. Pract.*, vol. 20, no. 4, pp. 328–342, 2012.

[3] M. Scandella, A. Ghosh, M. Bin, and T. Parisini, "Traffic-light control in urban environment exploiting drivers' reaction to the expected red lights duration," *Transp. Res. C: Emerg. Technol.*, vol. 145, p. 103910, 2022.

[4] A. Ghosh, M. Scandella, M. Bin, and T. Parisini, "Traffic-light control at urban intersections using expected waiting-time information," in *Proc. 60th IEEE Conf. Decis. Control (CDC)*, 2021, pp. 1953–1959.

[5] B. Sonzogni, J. M. Manzano, M. Polver, F. Previdi, and A. Ferramosca, "CHoKI-based MPC for blood glucose regulation in artificial pancreas," in *Proc. 22nd IFAC World Congr.*, vol. 56, no. 2, 2023, pp. 9672–9677.

[6] G. Pozzato, M. Müller, S. Formentin, and S. M. Savaresi, "Economic MPC for online least costly energy management of hybrid electric vehicles," *Control Eng. Pract.*, vol. 102, p. 104534, 2020.

[7] L. Ljung, *System Identification: Theory for the User*. Prentice Hall, 1999.

[8] D. Piga, M. Forgione, S. Formentin, and A. Bemporad, "Performance-oriented model learning for data-driven MPC design," *IEEE Control Syst. Lett.*, vol. 3, no. 3, pp. 577–582, 2019.

[9] L. Hewing, K. P. Wabersich, M. Menner, and M. N. Zeilinger, "Learning-based model predictive control: Toward safe learning in control," *Annu. Rev. Control Robot. Auton. Syst.*, vol. 3, pp. 269–296, 2020.

[10] S. Formentin, K. Van Heusden, and A. Karimi, "A comparison of model-based and data-driven controller tuning," *Int. J. Adapt. Control Signal Process.*, vol. 28, no. 10, pp. 882–897, 2014.

[11] V. Krishnan and F. Pasqualetti, "On direct vs indirect data-driven predictive control," in *Proc. 60th IEEE Conf. Decis. Control (CDC)*. IEEE, 2021, pp. 736–741.

[12] J. Berberich, J. Köhler, M. A. Müller, and F. Allgöwer, "Data-driven model predictive control with stability and robustness guarantees," *IEEE Trans. Autom. Control*, vol. 66, no. 4, pp. 1702–1717, 2020.

[13] N. Ab Azar, A. Shahmansoorian, and M. Davoudi, "From inverse optimal control to inverse reinforcement learning: A historical review," *Annu. Rev. Control*, vol. 50, pp. 119–138, 2020.

[14] W. Edwards, G. Tang, G. Mamakoukas, T. Murphey, and K. Hauser, "Automatic tuning for data-driven model predictive control," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*. IEEE, 2021, pp. 7379–7385.

[15] F. Abbracciavento, F. Zinnari, S. Formentin, A. G. Bianchessi, and S. M. Savaresi, "Multi-intersection traffic signal control: A decentralized MPC-based approach," *IFAC J. Syst. Control*, vol. 23, p. 100214, 2023.

[16] A. Chiuso and G. Pillonetto, "System identification: A machine learning perspective," *Annu. Rev. Control Robot. Auton. Syst.*, vol. 2, pp. 281–304, 2019.

[17] M. Mazzoleni, A. Chiuso, M. Scandella, S. Formentin, and F. Previdi, "Kernel-based system identification with manifold regularization: A bayesian perspective," *Automatica*, vol. 142, p. 110419, 2022.

[18] F. Dörfler, J. Coulson, and I. Markovsky, "Bridging direct and indirect data-driven control formulations via regularizations and relaxations," *IEEE Trans. Autom. Control*, vol. 68, no. 2, pp. 883–897, 2022.

[19] V. Breschi, A. Chiuso, and S. Formentin, "Data-driven predictive control in a stochastic setting: A unified framework," *Automatica*, vol. 152, p. 110961, 2023.

[20] G. Pillonetto and G. De Nicolao, "A new kernel-based approach for linear system identification," *Automatica*, vol. 46, no. 1, pp. 81–93, 2010.

[21] M. Scandella, M. Mazzoleni, S. Formentin, and F. Previdi, "Kernel-based identification of asymptotically stable continuous-time linear dynamical systems," *Int. J. Control*, vol. 95, no. 6, pp. 1668–1681, 2021.

[22] T. Chen, "On kernel design for regularized LTI system identification," *Automatica*, vol. 90, pp. 109–122, 2018.

[23] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos, "The explicit linear quadratic regulator for constrained systems," *Automatica*, vol. 38, no. 1, pp. 3–20, 2002.

[24] J. Coulson, J. Lygeros, and F. Dörfler, "Regularized and distributionally robust data-enabled predictive control," in *Proc. IEEE 58th Conf. Decis. Control (CDC)*. IEEE, 2019, pp. 2696–2701.

[25] ——, "Data-enabled predictive control: In the shallows of the DeePC," in *Proc. 18th Eur. Control Conf. (ECC)*. IEEE, 2019, pp. 307–312.

[26] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. MIT Press Ltd, 2006.